3. Oleg

Program Name: Oleg.java Input File: oleg.dat

Oleg loves board games, especially chess and tic-tac-toe. After seeing an old Star Trek episode showing Spock and his 3D chess set, he thought about writing an AI program to play 3D chess, but decided to start with something easier, like 3D tic tac toe.

However, he needs your help to write this program. He has decided to input a string of characters to represent the three boards, like this one, "`O-O--X-XOO-X-X-OO-X-X-X---O`". Each level has 3 rows and 3 columns, as you can see in the diagram below: the first board with positions numbered 1 through 9, the second 10 through 18, and the third 19 through 27, with the entire diagram representing the given string.

1 O	2 -	3 O
4 -	5 -	6 X
7 -	8 X	9 O

10 O	11 -	12 X
13 -	14 X	15 -
16 O	17 O	18 -

19 X	20 -	21 X
22 -	23 X	24 -
25 -	26 -	27 O

To start with, he just wants to know, given a game already underway, if an additional move by X will win the game.

For example, the display shows the current moves for X and O, with the string "`O-O--X-XOO-X-X-OO-X-X-X---O`" representing those moves. If X plays on position 5 (row 2, col 2 of the top board), will X win the game? The answer is yes, since positions 14 and 23 immediately below position 5 already have Xs on them, winning with three Xs in a row. In fact, any "three in a row" answer will win, whether it vertical, horizontal, or diagonal on the original board, or some series that is in a straight line on all three boards. For example, if an X had been played in position 20, the winning series would be 8, 14, and 20.

Input - Several lines of data, each with two items: a 27 character string containing X, O and - indicating played and open positions on the three boards, followed by an integer representing the next move by X.

Output - All of the possible wins for X with this last move, showing all three position values (in ascending sorted order) that cause the win, or the words "NO WIN" if there is no win with this move. If the move results in several possible "wins", list each combination in order by 1st number, then if necessary by second number. A blank line will follow the output result for each data set.

Sample data:

```
O-O--X-XOO-X-X-OO-X-X-X---O 5
XO--X-O--O---X-XXO--X---O-O 11
X-X-X-O-OO-O-O-X-XX-X---O-O 23
```

Sample Output:

```
5 14 23

1 11 21
11 14 17

NO WIN
```